Bits & Software Magazine (2017) 1(1): 14-20.
ISSN: En trámite

Análisis de imágenes con OpenCV empleando interfaces en java

J. G. García Tovar, J. U. Romero López

***Resumen*. Las proteínas expresadas en un gel de poliacrilamida se identifican por manchas que pueden traslaparse y que a simple vista no siempre puede identificarse si alguna proteína de interés biológico se encuentra presente o no en el análisis biomédico o biotecnológico. Por ello, utilizando el IDE blueJ, junto con java y las librerías de OpenCV se generaron interfaces que permiten dar el procesamiento inicial de imágenes de geles de proteínas obtenidos por electroforesis de geles de poliacrilamida (SDS-PAGE) teñidos con azul de Coomassie y permitir la separación de las bandas lo suficiente para identificar la presencia o ausencia de la proteína de interés.**

***Palabras Clave*. *Análisis de imágenes, gel de poliacrilamida, OpenCV, java, blueJ*.**

***Abstract*. The proteins expressed in a polyacrylamide gel are identified by overlapping spots and sometimes are not visible to the naked eye if a protein of biotechnology interest was or not expressed in the protein studies. For that, using blueJ IDE with java and OpenCV libraries we make software interfaces for the preliminary images analysis of gels of proteins obtained with polyacrylamide analysis gel electrophoresis (SDS-PAGE) stained with Coomassie blue and separate enough the spots for identify the presence or absence of protein of the interest.**

***Keywords. Image analysis, polyacrylamide gel, OpenCV, java, blueJ*.**

I. INTRODUCCIÓN

El análisis y el procesamiento de imágenes proporcionan un medio para extraer y cuantificar objetos y patrones a partir de los datos de una imagen y obtener respuestas a preguntas significativas no solamente en el área de las ciencias computacionales, sino también en biomedicina. La segmentación de imágenes es el proceso de dividir los elementos de una imagen en grupos de tal modo que todos los elementos en un grupo tienen una propiedad en común. [1-3].

José Guadalupe García Tovar.
Universidad Politécnica Metropolitana de Puebla.
jose_garcia_tovar@hotmail.com

Jesús Uriel Romero López
Universidad Politécnica Metropolitana de Puebla.
jesus.uriel.romero.lopez@gmail.com

En el dominio biomédico, ésta propiedad común usualmente se refiere a los elementos que pertenecen al mismo tejido celular u órgano. La segmentación de las estructuras anatómicas es una tecnología clave que asiste al médico o analista en el diagnóstico, planificación y orientación del paciente. La segmentación permite la visualización de las estructuras de interés así como remover información innecesaria de tal manera que se convierte en un proceso fundamental en la mayoría de los sistemas que soportan diagnóstico médico. El proceso de segmentación es un paso previo para el éxito de diversas aplicaciones tales como lesiones traumáticas cerebrales, extracción automática de perfiles de DNA o análisis de proteínas expresadas en una muestra de una población específica [4,5].

Los avances en la investigación biomédica y en biología molecular dependen de la interpretación de los datos obtenidos durante los experimentos realizados en la expresión de genes o de proteínas, estos resultados son analizados mediante geles de agarosa o de poliacrilamida que consisten en la representación de bandas ubicadas en una fila y columna específica. Una interpretación incorrecta de estas bandas lleva a conclusiones erróneas. Por otro

lado, un gel con bandas borrosas o muy decolorado son descartados o eliminados y se tienen que repetir los experimentos, lo que lleva al problema de no poder recolectar otra muestra si no se cuenta con la célula, el gen o el individuo que requiere dicho análisis [5].

En 1999 se publicó por parte del equipo de Xiangyun Ye el primer acercamiento hacia como procesar un gel de poliacrilamida empleando transformadas Top Hat después de un análisis de picos del histograma de la imagen para poder detectar la existencia o ausencia de bandas en una región de un gel [6]. Para el 2000, Alon Efrat plantea el uso de algoritmos geométricos para poder identificar una mancha que sea similar entre dos imágenes diferentes [7].

Por otro lado, Ivan Bajla en el 2001 presenta un análisis de gel de ADN logrando eliminación de ruido de fondo del gel empleando una convolución Gaussiana 1D [8].

En el 2010, el equipo de Andrew W. Dowsey publica una comparación entre los diferentes programas comerciales que existen para poder eliminar ruido en la imagen de los geles y que permitan alinear las bandas que representan a las proteínas mejorando la calidad de la imagen y que pueda ser interpretada por el analista [9].

En el 2013 Koprowski y su equipo publican el que hasta el momento es el mejor trabajo en cuanto al análisis de geles de poliacrilamida. Demuestran que los programas disponibles no pueden lograr la identificación de las bandas de un gel. Utilizan imágenes de alta calidad obtenidas con un fotodocumentador alcanzando una resolución de 3806x1027 pixeles. Las técnicas empleadas en su investigación permiten obtener imágenes corregidas y que puedan ser fácilmente interpretadas por los analistas del departamento de genética de la universidad de Silesia en Polonia [5].

Jan M. Brauner en el 2014 consigue un algoritmo que permite hacer correlaciones de las bandas que contiene el gel de poliacrilamida y después de tratar la imagen permite eliminar falsos positivos y que el analista pueda diferenciarlos de las bandas que corresponden a las proteínas expresadas reales [10].

El software desarrollado permite la reducción de experimentos y facilita la interpretación en el gel buscando si se ha expresado o no la proteína que el analista está buscando, sin que esta dependa de la baja decoloración de la imagen evitando que el gel sea eliminado o que los resultados obtenidos sean descartados sin tener que repetir experimentos que en los laboratorios de biomedicina resultan costosos.

OpenCV

OpenCV es el acrónimo de Open Source Computer Vision, es una librería de software libre, de código abierto que permite facilitar el análisis de imágenes y el aprendizaje computacional. Fue cosntruido para proporcionar una infraestructura común para apllicaciones de visión computacional y acelerar el uso de las computadoras hacia productos comerciales. Empezó como un producto licenciado por BSD y que evolucionó hacia software de código abierto para poder utilizarse en negocios y permite realizar modificaciones en el código [11].

OpenCV es una librería con mas de 2500 algoritmos optimizados, utilizada por empresas importantes y que además puede ejecutarse en sistemas operativos windows, linux/unix, Android y Mac OS. Existen instrucciones escritas para poder ser ejecutadas desde diferentes lenguajes de programación como C++, C, Python, Java y MATLAB.

Java

Java es un lenguaje de programación que tomó gran auge con el internet, es un lenguaje confiable, rápido y seguro. La ventaja de java es que los programas escritos en este lenguaje no necesitan ser modificados para ser ejecutados en otros sistemas operativos, basta con tener instalada la máquina virtual de java y poder utilizarla en diferentes plataformas y el programa funcionará sin modificaciones, inicialmente le pertenecía a Sun microsystems y actualmente lo distribuye Oracle. Es el estándar global para desarrollar y distribuir aplicaciones móviles y desarrollo de juegos [12].

BlueJ

Dentro de los ambientes de desarrollo de java existen Netbeans, Eclipse y BlueJ. Este último tiene la ventaja de no requerir muchos recursos de hardware para el desarrollo de aplicaciones java, presenta una interfaz muy sencilla para que los programadores puedan desarrollar software con la desventaja de que la forma de integrar la información la realiza el programador, es decir, las interfaces gráficas de java

tienen que ser definidas por el programador y no por BlueJ. La ventaja de hacer esto es que el usuario tiene el control de todo el código y un IDE que facilita la programación, el compilado y la ejecución de sus programas. Otra ventaja muy importante que tiene BlueJ es que permite integrar librerías externas de tipo JAR de manera muy simple, basta con agregarlas en el menú de herramientas y preferencias y BlueJ reconocerá el código de las nuevas librerías como es el caso de requerir integrar las librerías javacpp.jar, javacv-windows-x86.jar, javacv.jar y open-245.jar. BlueJ es desarrollado por la Universidad de Kent y avalado por Orale [13].

II. METODOLOGÍA EXPERIMENTAL

Para realizar un estudio sobre expresión de proteína primero se requiere agregar la secuencia genética de una proteína recombinante dentro de un vector de expresión bacteriano, insertarlo dentro de la bacteria, crecerla en un medio de cultivo controlado e inducir su expresión con algún activador. Posteriormente, se realiza algún método de purificación de proteínas y se realiza la separación de las mismas por SDS-PAGE (electroforesis de gel de poliacrilamida) para visualizarlas con el método de tinción de Coomassie (ver figuras 1 y 2).

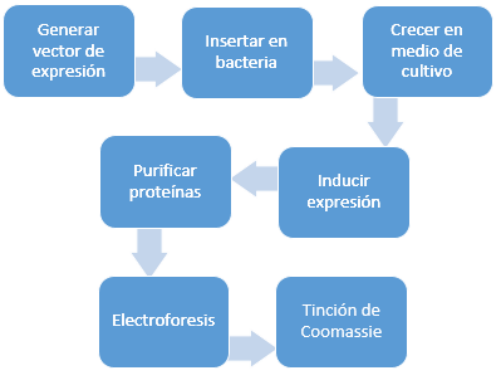

Figura 1. Diagrama de los pasos a seguir en la metodología para obtener geles de proteína

III. DISEÑO DEL SOFTWARE

Se diseñaron interfaces en java para poder realizar el análisis de imágenes preliminar que permita delimitar los contornos de las imágenes obtenidas de geles de poliacrilamida.

Se agregaron las librerías de OpenCV y se programaron para poder realizar la conversión de grises, el manejo de cambios en el umbral, se aplicaron las transformadas gaussianas y el filtro de imagen adaptativo. Se colocó un slider para que el usuario pueda manipular las opciones instaladas, y finalmente un menú para que se puedan abrir y guardar las imágenes generadas con el programa. La figura 3 muestra la interfaz inicial y la figura 4 el diagrama de los métodos generados para poder realizar el análisis de imágenes preliminar.

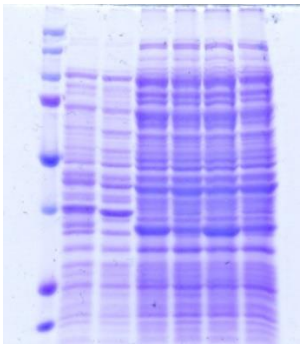

Figura 2. Gel de electroforesis de proteína recombinante expresada en bacterias usando el método de tinción de Coomassie.

Figura 3. Interfaz inicial donde el usuario incorpora la imagen para realizar el análisis de la misma

IV. RESULTADOS

A continuación se presentan los resultados obtenidos con los experimentos en el laboratorio y con la aplicación del software.

RESULTADOS EXPERIMENTALES

La figura 5 muestra un ejemplo de los geles obtenidos después de realizar la purificación de proteínas recombinantes y que se usaron en el dataset de imágenes para probar el software desarrollado. La tinción se realizó con el método de azul de

Coomassie y la imagen se capturó con un escaneo del gel de electroforesis obtenido.

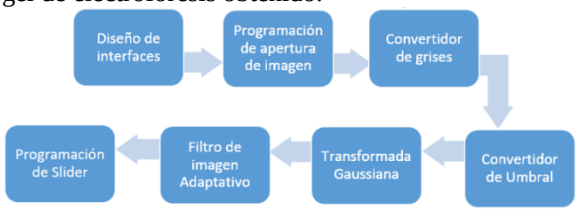

Figura 4. Diagrama de los métodos programados para tener el software que permite realizar el análisis de imágenes de geles de proteínas.

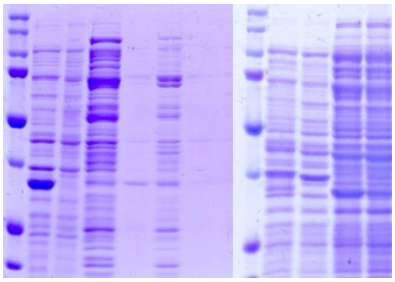

Figura 5. Geles de poliacrilamida obtenidos de la purificación de proteína recombinante expresada en bacterias reveladas con azul de Coomassie

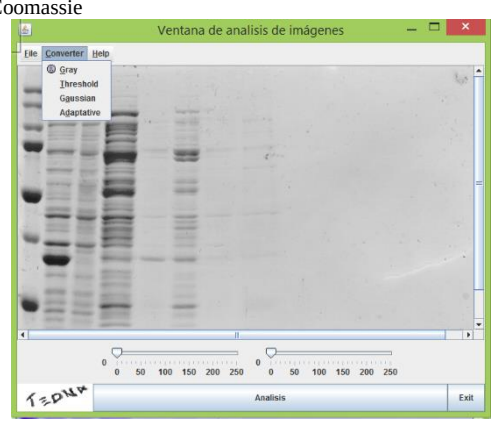

Figura 6. Conversión de la imagen a grises

RESULTADOS DEL ANÁLISIS DE IMÁGENES PRELIMINAR

Para poder realizar la programación de las interfaces empleando el convertidor de grises, el manejo de umbral, la transformada gaussiana, el filtro de imagen adaptativo con las librerías de OpenCV, se buscaron las versiones que pudieran funcionar adecuadamente con el lenguaje de programación Java junto con el IDE BlueJ. Después de probar diferentes versiones las que dieron la funcionalidad requerida fueron:

-OpenCV (versión 2.4.5).
-BlueJ (versión 3.1.7).
-Java (versión JDK 6.21 o superior).

El primer análisis consiste en convertir la imagen a grises para eliminar la tonalidad del azul de Coomassie, éste primer paso permite al analista que pueda identificar la tonalidad de las bandas y apreciar si existe alguna con mayor intensidad que otra para verificar si se ha expresado o no la proteína recombinante (figura 6).

El rango de la escala de niveles de grises de los objetos se obtiene observando un perfil a través de la imagen. El nivel de fondo se coloca en cero (Z1) y el color de mayor intensidad se le asigna un valor máximo de gris (Z2). El resto de los colores se ajusta al gris con diferente tonalidad entre los dos valores mínimo Z1 y máximo Z2.

Si tomamos como B a la resolución en tonos de la imagen digitalizada, f(x,y) es la imagen a procesar y g(x,y) es la imagen resultante en grises entonces los colores pueden escalarse en 2^B-1 niveles (1):

$$g(x,y)=\begin{cases}\dfrac{(f(x,y)-Z_1)(2^B-1)}{Z_2-Z_1} & Z_1<f(x,y)<Z_2\\ 2^B-1 & f(x,y)\geq Z_2\\ 0 & f(x,y)\leq Z_1\end{cases}$$

En openCV se expresa como sigue:

IplImage src=cvLoadImage(ruta); //ruta de la imagen a analizar

CvSize cvSize=cvSize(src.width(),src.height()); //Tamaño de la imagen

IplImage gray=cvCreateImage(cvSize,src.depth(),1);//imagen inicial f(x,y)

cvCvtColor(src,gray,CV_BGR2GRAY);//conversión a grises

cvSaveImage("new.png",gray);//imagen de salida g(x,y)

Al aplicar un filtro de imagen adaptativa permite realzar el procesamiento o reestablecer los datos que han sido modificados, también permite delimitar los contornos de la imagen y con ello poder identificar si existe alguna diferencia en la imagen (figura 7).

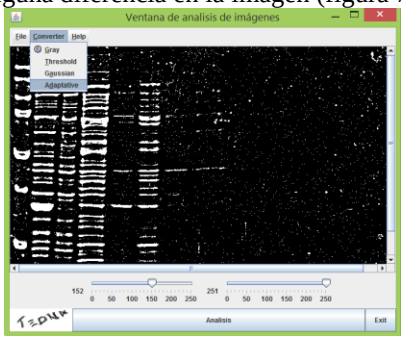

Figura 7. Filtro de imagen adaptativa para resaltar contornos de la imagen.

El filtro adaptativo puede detectar los bordes de cada pixel por que a cada elemento de la imagen de entrada f(x,y) le asigna una matriz A cambiando los valores a su alrededor como lo muestra la tabla 1.

Tabla 1. Matriz de valores para el filtro adaptativo de cada pixel.

A_0	A_1	A_2
A_7	$f(i,j)$	A_3
A_6	A_5	A_4

Por conveniencia representamos a (2)

$$A_i = A_{i+8} \quad (2)$$

Donde tenemos que (3,4)

$$S_i = A_i + A_{i+1} + A_{i+2} \quad (3)$$

$$T_i = A_{i+3} + A_{i+4} + A_{i+5} + A_{i+6} + A_{i+7} \quad (4)$$

Que nos permite obtener la imagen de salida $g(x,y)$ definida como (5):

$$g(x,y) = \max\left(1, {}_{i=0}^{7}max(|5S_i - 3T_i|)\right) \quad (5)$$

En OpenCV se expresa como sigue:

IplImage src=cvLoadImage(ruta);//imagen de entrada f(x,y)
CvSize cvSize=cvSize(src.width(),src.height());//cálculo de tamaño
IplImage gray=cvCreateImage(cvSize,src.depth(),1);//ajuste de tamaño
cvCvtColor(src,gray,CV_BGR2GRAY);//conversión a grises
cvAdaptiveThreshold(gray, gray, 255, CV_ADAPTIVE_THRESH_MEAN_C, CV_THRESH_BINARY_INV, 11, 5);//filtro adaptativo
cvSaveImage("new.png",gray);//imagen de salida g(x,y)

Para detectar si una proteína recombinante se ha expresado, el gel de poliacrilamida se divide en columnas, donde cada una de ellas presenta diferentes experimentos realizados por el analista.

La detección de una proteína recombinante consiste en identificar si existe una banda que no aparezca en ambas columnas, la cual no siempre es visible a simple vista como en la figura 8.

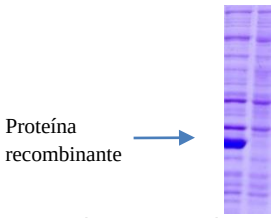

Figura 8. Detección de la proteína recombinante en el gel de poliacrilamida.

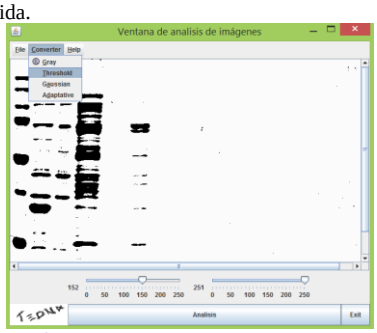

Figura 9. Análisis gaussiano de la imagen del gel de poliacrilamida.

Si el análisis realizado hasta el momento no permite la detección de la proteína recombinante expresada, se puede realizar la transformada gaussiana para delimitar el contorno de las bandas (figura 9)

El filtro gaussiano permite colocar una estructura gaussiana a los datos de la imagen. Es decir, resaltar más el centro del pixel y disminuir la intensidad de los mismos en sus alrededores. Esto permite que la imagen pueda tener puntos de resalte y eliminar el contorno de la misma que en algunas ocasiones se considera como ruido.

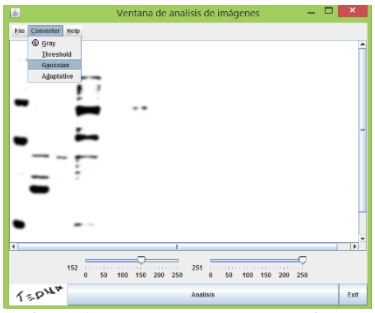

Figura 10. Análisis final de la imagen después de modificar el umbral.

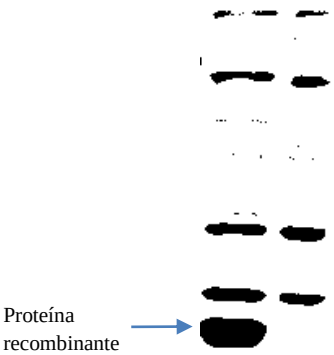

Figura 11. Imagen final obtenida después de realizar el análisis preliminar con el programa desarrollado.

Primero se requiere de tener la imagen digitalizada f(x,y) que es obtenida en grises tal y como se expresó en la ecuación 1 y después realizar la conversión mediante la función

$$g(x,y) = e^{\frac{-(x+y)^2}{2\sigma^2}} \quad (6)$$

donde (x,y) son las coordenadas de la imagen y sigma (σ) es la desviación estándar de la probabilidad de distribución asociada.

En OpenCV queda como:

IplImage src=cvLoadImage(ruta);//ruta de la imagen
CvSize cvSize=cvSize(src.width(),src.height());//obtención de ancho y alto
IplImage gray=cvCreateImage(cvSize,src.depth(),1);//calculo de la profundidad
cvCvtColor(src,gray,CV_BGR2GRAY);//conversión en grises
cvSmooth(gray, gray, CV_GAUSSIAN, 13); //filtro gaussiano
cvSaveImage("new.png",gray); //imagen de salida f(x,y)

y por último, modificando el umbral de la imagen el analista puede identificar la banda buscada y al final de todo el proceso descrito anteriormente se pueda identificar la posición de la proteína y verificar su expresión.

OpenCV permite manejar el umbral para blancos y el umbral para negros. Para ello, sea τ el valor proporcionado de umbral (blanco o negro) de una imagen digitalizada proporcionada por el usuario en diferentes niveles de grises, B es el número de bits en la imagen, entonces (7)

$$G = \{0,1,2,\dots,2^B\} \qquad (7)$$

G corresponde a los diferentes niveles de grises y (8)

$$C = \{C_0, C_1\} \qquad (8)$$

C contiene los elementos que corresponden al par de niveles de gris binario, lo que nos proporciona la siguiente imagen como salida en el cambio de umbral (9)

$$g(x,y) = \begin{cases} C_0, & si\ f(x,y) < \tau \\ C_1, & si\ f(x,y) \geq \tau \end{cases} \qquad (9)$$

El código en OpenCV es:

```
IplImage src=cvLoadImage(ruta);//ruta de la imagen f(x,y)
CvSize cvSize=cvSize(src.width(),src.height());//tamaño de la imagen
IplImage gray=cvCreateImage(cvSize,src.depth(),1);//profundidad de la imagen
cvCvtColor(src,gray,CV_BGR2GRAY);//conversión a grises
cvThreshold(gray, gray, evaluo,evaluo2, CV_THRESH_BINARY);
//cálculo del umbral blanco y negro () tomando los valores proporcionados por los slider
cvSaveImage("new.png",gray);//imagen de salida g(x,y)
```

Como lo muestra la figura 11, después de haber realizado todo el análisis de imágenes y de haber aplicado los métodos mencionados en la figura 4, puede verse que las columnas presentan las mismas bandas a excepción de una mancha que aparece con mayor intensidad en la primera columna, que no aparece en la segunda y que corresponde a la proteína buscada por el analista en el gel permitiendo que todo este análisis preliminar pueda identificar que se ha expresado la proteína recombinante en bacterias y que el uso del software permite facilitar el análisis de los geles de poliacrilamida.

Además, puede apreciarse que el uso de las librerías de OpenCV facilita la programación al momento de realizar los cálculos de umbral, el manejo de filtros gaussianos o adaptativos y que es una herramienta que facilita el análisis de imágenes y puede ser utilizado en cualquier lenguaje de programación.

V. CONCLUSIONES

El análisis de imágenes biomédicas es un área que va en desarrollo debido a su importancia para el diagnóstico de enfermedades. El uso de librerías de OpenCV facilita la programación y el análisis de las mismas. La integración entre Java, OpenCV y el IDE BlueJ permite realizar interfaces donde el usuario tiene un mayor control en el análisis de sus imágenes lo que no permiten otros lenguajes.

VI. AGRADECIMIENTOS

Al cuerpo académico de Biotecnología y Química Aplicada de la Universidad Politécnica Metropolitana de Puebla por el material recibido de los experimentos realizados para la obtención de los geles de poliacrilamida de proteínas.

VII. REFERENCIAS

[1] Sossa Azuela JH, García Tome A. (2011). Procesamiento y Análisis Digital de Imágenes. España, ES, RA-MA S.A. Editorial y publicaciones.

[2] Martín Roldán MA. (2012). Análisis DAFO de metadatos en imágenes digitales. España, ES, Editorial Académica Española.

[3] González R, Woods R. (1993). Digital Image Processing. USA, Addison-Wesley.

[4] Deserno TM. (2011). Biomedical Image Proocessing. USA. Springer.

[5] Koprowski R. Wróbel Z. Korzy´nka A. Chwialkowska K. Kwasniewski M. (2013). Automatic analysis of 2D polyacrylamide gels in the diagnosis of DNA polymorphisms. Biomedical Engineering OnLine, 1(12), 1-14.

[6] Ye X, Suen CY, Cheriet M, Wang E. (1999). A recent Development in Image Analysis of Electrophoresis Gels. Vision Interface. Canada, May 19-21.

[7] Efrat A, Hoffmann F, Kriegel K, Schultz C, Wenk C. (2001). Geometric algorithms for the analysis of 2D-Electrophoresis gels. RECOMB 01, Proceedings of the fifth annual international

conference on computational biology. USA. 114-123.

[8] Bajla I, Holländer I, Burg K. (2001). Improvement of electrophoretic gel image analysis. Measurement Science Review. 1 (1) 5-10.

[9] Dowsey A, Morris JS, Gutsein HB, Yang GZ. (2010). Informatics and statistics for analyzing 2-D gel electrophoresis images. Methods Mol. Biol. 1 (604) 239-255.

[10] Brauner JM, Groemer TW, Stroebel A, Grosse-Holz S, Oberstein T, Wiltfang J, Kornhuber J, Maler JM. (2014). Spot quantification in two dimensional gel electrophoresis image analysis: comparison of different approaches and presentation of a novel compound fitting algorithm. Bioinformatics. 15 (181) 112.

[11] Itzeez. (2016) http://www.opencv.org/about.html

[12] java (2016) https://www.java.com/es/about/

[13] BlueJ (2016) http://www.bluej.org/index.html

VIII. BIOGRAFÍA

José Guadalupe García Tovar es alumno del último cuatrimestre de la Ingeniería en Sistemas Computacionales de la Universidad Politécnica Metropolitana de Puebla.

Jesús Uriel Romero López es alumno del último cuatrimestre de la Ingeniería en Sistemas Computacionales de la Universidad Politécnica Metropolitana de Puebla.